Linear Tape File System Single Drive Edition (LTFS SDE) Best Practices

Content obtained by permission from "Optimizing LTFS Usage" whitepaper by Shawn O. Brume, Sc.D, IBM, United States

Abstract

With the introduction of the Linear Tape File System (LTFS), a new dimension of capability and usability has been opened up for the tape storage environments. While the usability of the tape drive in general is improved to the level of using a USB flash drive, the misconception that the tape drive storage on LTFS will work exactly like a disk drive is not completely accurate. This white paper will help to clarify some of the misconceptions and introduce operational considerations not yet considered.

LTFS Definition

LTFS is a self-describing capability for tape drives that is enabled by partitioning. The tape partitioning allows an XML index to be stored on a small partition for fast access and ease of update and the second partition to contain data files. LTFS is supported on LTO5 and LTO6 tape drives only. See figure 1 below for the partition structure.

Figure 1 – LTFS Partitions

Files are saved to the tape and corresponding metadata about the files are also written to tape. See Figure 2 below for a simple 3 file example.

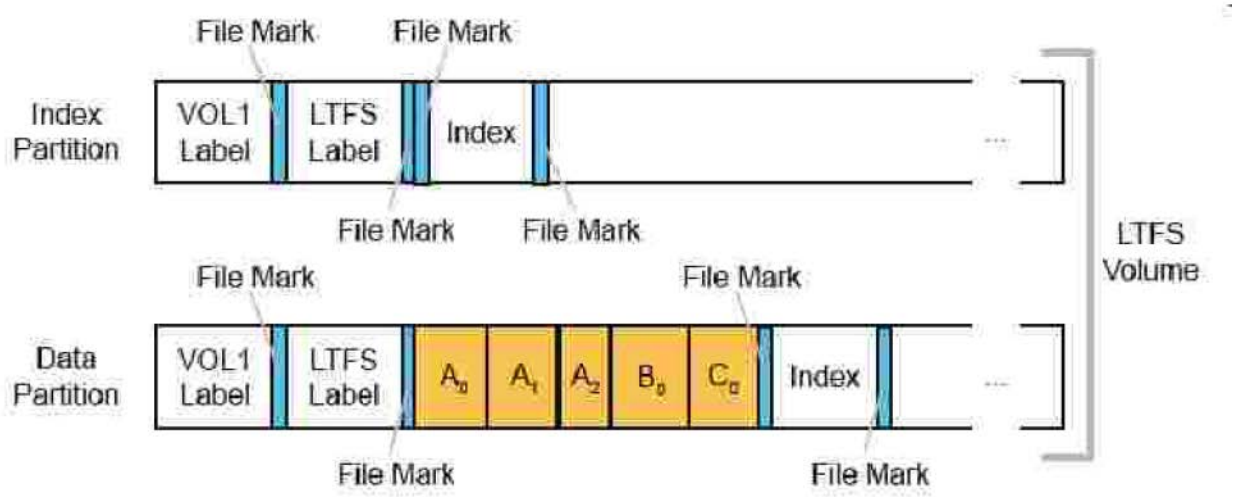

Figure 2 – Simple 3 File Example

Files can be saved, modified, deleted and moved to new directories in the file system. Modifications and deletions do not actually remove or change the files on tape, these files become versions on the tape allowing the consumer to back level at a later date, if needed. All of this can be achieved through the GUI file manager of the OS or via a command prompt.

Using the LTFS SDE software, tapes are able to be loaded to tape drives and presented to the host as a mount point. In Microsoft Windows© that mount point is a drive letter, in Linux it is a mount point similar to any other mount point. In the most basic form LTFS displays no files when a tape is not loaded to the tape drive, this is the same as the operation of a USB flash drive behavior. Files can be manipulated in the same manner that they are manipulated on a disk or flash drive. See figure 3 below.

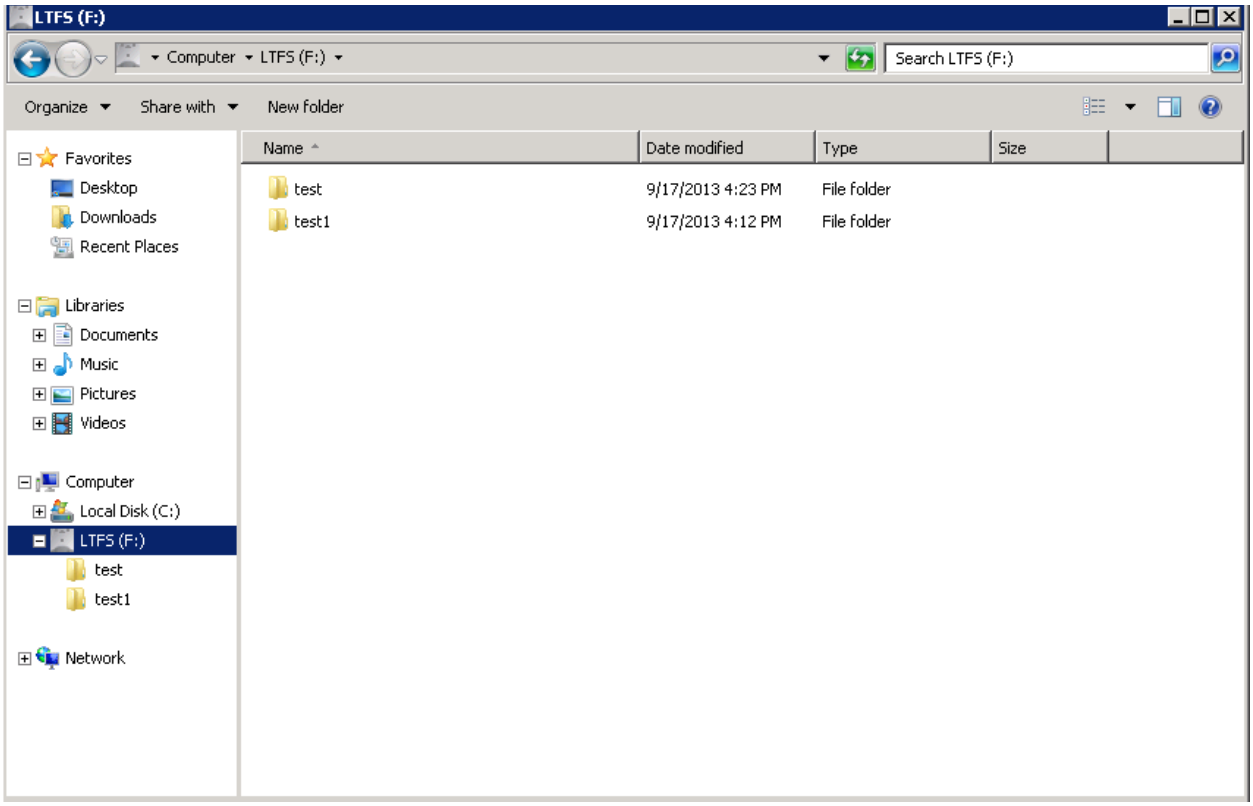

Figure 3 – LTFS SDE Directory Structure

Files written to tape on one platform, host OS or LTFS implementation, are readable and cartridges are fully usable on other platforms.

The innovations of LTFS allow the tape drive to be used with the ease of functionality that disk and flash drives are used. The important point is that the tape drive is still a linear (sequential) device, placing data in a linear fashion along the length of wraps of the tape. The LTFS driver has been developed to allow the response to the user interface to be similar to that of non-linear storage devices like disk and USB drives.

Operating systems can perform more than 1 task at a time, including multiple tasks or actions to a single device in many cases. The capability of an operating system to execute multiple actions at the same time is called multi-threading. An example of a multi-thread action is to drag-and-drop a large file to a storage device and then drag-and-drop a second file to the same storage device. Both copies are executed at the same time to the storage device rather than 1 completing then the second being started (single thread). Multi-thread actions can be accomplished through programming in an application, executing multiple scripts at the same time, or performing multiple, separate drag-and-drops at the same time. When dragging and dropping multiple files selected at the same time, for instance an entire directory is copied with a single drag-and-drop motion, the files are saved in a single threaded manner, each file transfer must complete before the next begins.

LTFS allows multi-thread copies to and from the tape and saves the files in an interlaced manner. Interlacing is the multi-thread capability of LTFS that allows the host system to save data continuously from more than one save action. The resulting data on tape is called interlaced and in contrast to the clean data blocks on tape demonstrated in Figure 2, the files are now in smaller separated chunks that can be spread across long areas of tape depending on the files size interlaced to the tape. Figure 4 shows the interlaced files as they look logically on tape.

Figure 4 – Interlaced Files

Reads and Writes Using LTFS SDE

The time to gain access to the actual files in the file system is dependent on the location of the data on the tape media and the time needed to seek to that portion of the media and read the data.

Contiguous writes are file writes that are performed in a sequential manner. Sequential writes are writes done in a single system thread. As an example, any copy command execution is a single write. The copy command can be done as a single command to copy entire directories or even multiple directories. It is not limited to a single file.

This same method is employed on drag-and-drop operations. Any selection of multiple files or directories in a GUI environment and executed as a single drag is considered a single threaded copy.

Interlaced file writes are created when multiple threads are used simultaneously to write data to tape. Multithread writes can be issued using multiple copy commands at the same time, such as multiple command windows with copy commands issued from each or running batch copy files at the same time to the same tape.

Interlacing in a GUI environment is achieved by doing multiple drag-and-drop operations to the same tape simultaneously. This can be multiple or single files copied in

each drag-and-drop operation. The resulting file structure on the tape is shown in Figure 4 above.

It is recommended that multi-threaded writes be avoided as this can impact read performance in certain circumstances as outlined in the read performance section below.

File Read Performance

Read operations are performed at the highest negotiated transfer rate, for LTO6 this is up to a rate of 160 MB/s for non-compressed data. In the most optimal environments with contiguous files the transfer time of files from LTFS media will be near the negotiated data rate of the system with minimal read back impact for locating when copying back multiple files. Small transfers and the beginning of large file transfers will be slower due to the time it takes for the drive to spin up.

In the case of interlaced files, the impact to read can be greatly affected. The impact depends on the number of threads used to copy the data and the size of the files copied. Generally the larger the files the worse the interlacing and the worse the read back performance. A further explanation of the impact of interlacing is that for every file in a multithread write the file is broken into smaller and smaller pieces, with the number of threads dictating how many times the data is divided and by how much of a divide there is between each block of file data. In effect, if 10 movies were to be copied to a LTFS directory using individual drag-and-drop threads, each movie may end up being spread across the complete length of tape. Read of a single file will require hundreds of read-> locate-> pause sequences to retrieve the entire file.

File Access times from Mounted and Active State

File access times are not affected by the manner in which the files have been saved to tape (contiguous or interlaced). LTFS uses the location on tape indicated in the metadata, as a way to accelerate the positioning of the tape for read. LTFS positions the head in the required region of tape and issues a SCSI locate command to find the required file.

GUI File System tips

The graphical interfaces of most operating systems allow for display of the metadata related to the files in the file system. The degree at which the data is displayed is dictated by the user settings in the GUI. Certain settings can cause a slower response in the LTFS implementation and it is recommended that they not be used.

In all GUI displays it is recommended that the settings be set to display the filename only. It is highly recommended that the GUI specifically be set to not display icons as this will slow the response to a directory open in LTFS because LTFS must retrieve the icon being used for each file.

Several operating systems allow for limited data recovery when a file is deleted. This is achieved by copying data from the deleted file system to the trash folder. In the case of LTFS the file must be accessed and copied to the trash folder which can take a long time if the file is large. It is recommended that this feature be disabled as LTFS keeps a copy of the deleted file on the tape after it is deleted from the displayed file system. This LTFS feature can also be used to back level the file to a previous version if changes were made to a file and resaved.

Special Considerations

It is recommended that when using consumer applications, such as Microsoft Word©, Excel©, and media players, that files be copied to the local hard drive before usage. It is possible to use the files directly from the tape, however a performance degradation may be observed in larger systems as a result of the auto-save features in many consumer applications. Auto-save writes a temporary copy of the file in use. This temporary file creation can create a perception of performance degradation when rewriting the temporary file updates during usage and deleting the file when the application is closed.

Some media players may experience issues with the buffering feature for video as a result of the high performance of the tape system actually degrading to a point that the data stream is not sufficient for the media player. Media players that do not require a buffering (load an entire file) will not have this issue, but it is still recommended that the files be moved locally to the hard disk when the files are only used in a read-only manner.

File Naming Conventions for Optimal Interoperability

The IBM LTFS SDE software has been designed for maximum interoperability. As such a limitation on file naming conventions has been implemented to ensure file compatibility. This file compatibility limitation is not due to any limitation of the software itself but a designed architecture to ensure ease of data exchange.

The file naming convention used by LTFS is the combined limitation of the base OS design naming limitations. As such when naming files in any file system it is best to avoid special characters. See Table 1 for a short list of special characters to avoid.

Character	Name
/	Slash
\	Backslash
?	Question Mark
*	Asterisk
:	Colon
|	Vertical bar or pipe
"	Quote
<	Less than
>	Greater than

Table 1 – Special Characters to Avoid

In general a limitation exists in most files system which restricts a file name of no more than 255 characters including spaces and file extensions. This also applies to LTFS.

Conclusion

The invention of LTFS has made the usage of tape drives as convenient as disk and flash drives. The ability to copy data to and from media in an open specification without expensive proprietary software moves tape into an information exchange market that was previously unrealized.

The key to effective usage of LTFS is to keep in mind the minor limitations of the linear device. Tape drives stream data to and from the medium at data rates that far exceed the data rates of disk of comparable size and cost.

As an archive format, LTFS allows for the easiest data retrieval and no proprietary software is required to retrieve the data from tape. Any future operating system developed can systematically integrate LTFS support by allowing the drive interface to read the XML format on tape and display the file system for retrieval.

Customers desiring the flexibility of a flash drive, with the security and low cost of data storage of tape will find LTFS the most flexible of all solutions.